



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/678,207	09/20/2000	John V. Skinner JR.	GEMS8081.029	6319

27061 7590 03/28/2003

ZIOLKOWSKI PATENT SOLUTIONS GROUP, LLC (GEMS)
14135 NORTH CEDARBURG ROAD
MEQUON, WI 53097

EXAMINER

CAO, DIEM K

ART UNIT PAPER NUMBER

2126

DATE MAILED: 03/28/2003

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary

Application No.

09/678,207

Applicant(s)

SKINNER ET AL.

Examiner

Diem K Cao

Art Unit

2126

-- Th MAILING DATE of this communication app ars on th cover sheet with th correspondenc address --
Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 20 September 2000.
- 2a) ☐ This action is FINAL. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1-37 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 1-37 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☒ The drawing(s) filed on 20 September 2000 is/are: a) ☐ accepted or b) ☒ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
- 11) ☐ The proposed drawing correction filed on _____ is: a) ☐ approved b) ☐ disapproved by the Examiner.
If approved, corrected drawings are required in reply to this Office action.
- 12) ☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. §§ 119 and 120

- 13) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
a) ☐ All b) ☐ Some * c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
2. ☐ Certified copies of the priority documents have been received in Application No. _____.
3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).
* See the attached detailed Office action for a list of the certified copies not received.
- 14) ☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).
a) ☐ The translation of the foreign language provisional application has been received.
- 15) ☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

Attachment(s)

- 1) ☒ Notice of References Cited (PTO-892) 4) ☐ Interview Summary (PTO-413) Paper No(s). _____
- 2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948) 5) ☐ Notice of Informal Patent Application (PTO-152)
- 3) ☒ Information Disclosure Statement(s) (PTO-1449) Paper No(s) 4. 6) ☐ Other: _____

DETAILED ACTION

1. This Office Action is in response to the Application filed on 9/20/2000.
2. Claims 1-37 are presented for examination.

Drawings

3. The drawings are objected to as failing to comply with 37 CFR 1.84(p)(5) because they do not include the following reference sign(s) mentioned in the description: Fig. 1, reference number 29. A proposed drawing correction or corrected drawings are required in reply to the Office action to avoid abandonment of the application. The objection to the drawings will not be held in abeyance.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

5. Claims 1, 18-19, 31-33 are rejected under 35 U.S.C. 103(a) as being unpatentable over Admitted Prior Art (APA) in view of Davidson et al. (U.S. 5,630,136).

As to claim 1, APA teaches (pages 3-4) integrating an X Window visualization toolkit (Xt Intrinsic based visualization and graphics toolkit) with a JAVA application (Java application or applet), providing a JAVA application thread that includes a call to an X Window visualization toolkit (Java application requests a schedule camera position ... is rotated), and a JAVA process thread that comprises an X Window X event loop (X event loop).

Art Unit: 2126

However, APA does not explicitly teach suspending execution of the X event loop to prevent concurrency related data corruption while a call to the X Window visualization toolkit is made by the JAVA application thread. Davidson teaches only one thread is permitted to access the resource (toolkits; col. 1, lines 23-34 and Of the various threads ... unsafe resource; col. 5, line 52 – col. 7, line 23).

It would have been obvious to apply the teaching of Davidson to the system of APA because it provides a technique for serializing access to multithreading unsafe resource (summary of the invention).

As to claim 18, see rejection of claim 1 above. However, APA does not explicitly teach providing a plurality of JAVA application threads that each include a call to an X Window visualization toolkit or widget, selecting one of the plurality of application threads to execute and then suspending execution of the remainder of the plurality of application threads. It is well known in the art that a Java application is a multi-thread application, one of ordinary skill in the art would be able to modify the Java application to have multiple threads each makes a call to the X Window toolkit. Davidson teaches only one thread could access the resource, and the rest of threads are suspended.

As to claim 19, see rejection of claims 1 and 18 above.

As to claim 31, APA teaches the call to the X Window visualization toolkit or widget comprises a call to an X Window Intrinsics based toolkit or widget (Xt Intrinsic based visualization and graphics toolkit, Java application requests a schedule camera position ... is rotated; pages 3-4).

Art Unit: 2126

As to claim 32, APA teaches the X Window Intrinsics based toolkit comprises VTK or a toolkit based on VTK (X Window visualization ... VTK ... X Toolkit Intrinsics; pages 2-3).

As to claim 33, APA does not explicitly teach the call to the X Window visualization toolkit or widget is made using the JAVA Native Interface. It is well known in the art that Java application uses JNI to make call to native methods. It would have been obvious the call to the X Window toolkit is made using the JNI.

6. Claims 2, 20 and 23 are rejected under 35 U.S.C. 103(a) as being unpatentable over Admitted Prior Art (APA) in view of Davidson et al. (U.S. 5,630,136) further in view of Lee (Adding External Input Sources to the X Toolkit Event Loop).

As to claim 2, APA does not teach the X event loop comprises an X Window file descriptor function that coordinates an X event loop blocking read that is used to suspend execution of the X event loop. Lee teaches the X Toolkit's XtAppAddInput () function adds file descriptor input handling (Unix File Descriptor Input; page 2). It would have been obvious to one of ordinary skill in the art to apply the teaching of Lee to the system of APA because it would provide a method to add external input sources to the X Toolkit event loop.

As to claim 20, APA does not teach the X event loop performs a blocking read to suspend execution of the X event loop in step (c). Lee teaches the XtAppAddInput() function adds file descriptor input handling to the even loop by specify the file descriptor identifying the socket or pipe, the socket condition in which the application interested, and the callback function (Unix file descriptor input; page 2). It would have been obvious to apply the teaching of Lee to the system of APA because it provides a method to suspend a thread by condition.

As to claim 23, APA does not explicitly teach after step (b), an additional step comprising suspending execution of the one of the plurality of the application threads to allow the X event loop to finish processing any X event being processed by the X event loop before execution of the X even loop is suspended in step (c). APA teaches the X event loop makes call to the X Windows toolkit (also assume that the native ... dimensional scene; page 4), and Davidson teaches only one thread can access the toolkit (only the thread ... unsafe resource; col. 6, lines 25-37). It would have been obvious the application thread must be suspended while the X event loop makes call to the toolkit because it provides a method for serializing access to multithreading resource.

7. Claims 3-4, 21-22, 24-30, and 34-37 are rejected under 35 U.S.C. 103(a) as being unpatentable over Admitted Prior Art (APA) in view of Davidson et al. (U.S. 5,630,136) further in view of Orton et al. (U.S. 5,475,845) and Lee (Adding External Input Sources to the X Toolkit Event Loop).

As to claim 3, APA does not teach the application thread comprises a first write socket that is used to communicate a first data element to a first read socket of an XtAppAddInput file descriptor function of the X event loop, a read acknowledge socket of the application thread that acknowledges receipt of a second data element from a write acknowledge socket of the XtAppAddInput file descriptor function of the X event loop, and a second write socket that is used to communicate a third data element to a second read socket of the X event loop; and further comprising the steps of communicating a first data element from the first write socket of the JAVA application thread to the first read socket of the X event loop; performing an application thread blocking read that suspends execution of the application thread until a second

data element is discovered by the read acknowledge socket of the application thread; communicating the second data element from the write acknowledge socket of the X event loop to the read acknowledge socket of the application thread; resuming execution of the application thread when the second data element is discovered by the read acknowledge socket of the application thread; performing the X event loop blocking read that suspends execution of the X event loop until a third data element is discovered by the second read socket of the X event loop; making a call from the JAVA application thread to the visualization toolkit while execution of the X event loop is suspended; communicating the third data element from the second write socket of the application to the second read socket of the X event loop; and resuming execution of the X event loop when the third data element is discovered by the second read socket of the X event loop.

Lee teaches the `XtAppAddInput()` function adds file descriptor input handling to the event loop by specifying the file descriptor identifying the socket or pipe, the socket condition in which the application is interested, and the callback function (Unix file descriptor input; page 2). Orton teaches the functions `SendAndReceive` and `ReplyAndReceive` provide synchronization between client-side and server-side threads. It would have been obvious to apply the teaching of Lee and Orton to the system of APA because it would provide a method for synchronization between threads when accessing the same unsafe resource.

As to claim 4, APA does not teach the application thread further comprises a plurality of JAVA functions with a first one of the functions pausing execution of the X event loop by causing the first data element to be communicated to the first read socket of the X event loop and a second one of the functions resuming execution of the X event loop by causing the third data

element to be communicated to the second read socket of the X event loop. Lee teaches in multi-tasking operating systems like Unix, multiple programs share data or control through socket or pipe feature, and the `XtAppAddInput()` function adds file descriptor input handling to the even loop by specify the file descriptor identifying the socket or pipe, the socket condition in which the application interested, and the callback function (Unix file descriptor input; page 2). It would have been obvious to apply the teaching of Lee to the system of APA because it would provide a method for communication between Java application and X even loop.

As to claim 21, see rejection of claim 2 above.

As to claim 22, see rejection of claim 2 above..

As to claim 24, APA does not teach suspending execution of the one of the plurality of the application threads is accomplished by a blocking read that includes a read socket of the one of the plurality of application threads that receives a data element from a write socket of the X event loop. Lee teaches the `XtAppAddInput()` function adds file descriptor input handling to the even loop by specify the file descriptor identifying the socket or pipe, the socket condition in which the application interested, and the callback function (Unix file descriptor input; page 2). Also see rejection of claim 20 above.

As to claim 25, APA does not teach suspending execution of the X event loop in step (c) comprises (1) providing a pause function of the one of the plurality of the application threads that communicates a first data element to a first write socket that is linked to a first read socket of the X event loop and a process data element function of the X event loop that reads the first data element from the read socket and issues a blocking read; (2) communicating the first data element to the first write socket of the one of the plurality of the application threads; (3)

retrieving the first data element by the first read socket of the X event loop; and (4) invoking a blocking read suspending execution of the X event loop.

Lee teaches the `XtAppAddInput()` function adds file descriptor input handling to the even loop by specify the file descriptor identifying the socket or pipe, the socket condition in which the application interested, and the callback function (Unix file descriptor input; page 2). Orton teaches the functions `SendAndReceive` and `ReplyAndReceive` provide synchronization between client-side and server-side threads. It would have been obvious to apply the teaching of Lee and Orton to the system of APA because it would provide a method for synchronization between threads when accessing the same unsafe resource.

As to **claim 26**, APA does not teach resuming execution of the X event loop in step (e) comprises (1) providing a resume function of the one of the plurality of the application threads that communicates a second data element to a second write socket 5 that is linked to a second read socket of the X event loop; (2) communicating the second data element to the second write socket of the one of the plurality of the application threads; (3) retrieving the second data element by the second read socket of the X event loop; and (4) unblocking the blocking read resuming execution of the X event loop.

Lee teaches the `XtAppAddInput()` function adds file descriptor input handling to the even loop by specify the file descriptor identifying the socket or pipe, the socket condition in which the application interested, and the callback function (Unix file descriptor input; page 2). Orton teaches the functions `SendAndReceive` and `ReplyAndReceive` provide synchronization between client-side and server-side threads. It would have been obvious to apply the teaching of Lee and

Art Unit: 2126

Orton to the system of APA because it would provide a method for synchronization between threads when accessing the same unsafe resource.

As to **claim 27**, APA does not teach suspending execution of the X event loop in step (c) comprises (1) providing a pause function of the one of the plurality of the application threads that communicates a first data element to a first write socket that is linked to a first read socket of the X event loop, and a process data input function of the X event loop that reads the first data element from the read socket and communicates a second data element to write acknowledge socket that is linked to a read acknowledge socket of the one of the plurality of the application threads; (2) communicating the first data element to the first write socket of the 10 one of the plurality of the application threads; (3) invoking a first blocking read suspending execution of the one of the plurality of the application threads; (4) retrieving the first data element by the first read socket of the X event loop; (4) communicating the second data element to the write acknowledge socket of the X event loop; (5) retrieving the second data element by the read acknowledge socket of the one of the plurality of the application threads; (6) invoking a second blocking read suspending execution of the X event loop; and (7) unblocking the first blocking read resuming execution of the one of the plurality of the application threads. See rejection of claim 26 above.

As to **claim 28**, see rejection of claim 27 above.

As to **claim 29**, APA does not teach the process data input function comprises the XtAppAddInput function native to the X Window system. Lee teaches the XtAppAddInput() function adds file descriptor input handling to the even loop by specify the file descriptor identifying the socket or pipe, the socket condition in which the application interested, and the

Art Unit: 2126

callback function (Unix file descriptor input; page 2). It would have been obvious to apply the teaching of Lee to the system of APA because it would provide a method to add external function to the X event loop.

As to claim 30, APA teaches the process thread calls a native code function that serves as the X event loop (Xt Intrinsics based ...server states; pages 3-4).

As to claim 34, APA does not teach using an X Windows Intrinsics based visualization toolkit requiring an X event loop in a JAVA application, devoting a JAVA thread in the JAVA application to the X event loop; defining a write socket and a read acknowledge socket; calling XtAppAddInput to register a process input function on the write socket, wherein the process input function performs a processInput and read, a write acknowledge, a read, and a return X event loop; defining a pausing method that performs a write on the write socket, and a read acknowledge on the read acknowledge socket; defining a resuming method that performs a write on the write socket; and when a call is made to the visualization toolkit, first calling the pausing method, making the call to the toolkit, and then calling the resuming method.

See rejection of claims 1 and 3 above for rejection.

As to claim 35, see rejection of claims 1 and 3 above.

As to claim 36, see rejection of claims 1 and 3 above.

As to claim 37, see rejection of claim 2 above.

8. Claims 5-14 are rejected under 35 U.S.C. 103(a) as being unpatentable over Admitted Prior Art (APA) in view of Davidson et al. (U.S. 5,630,136), Orton et al. (U.S. 5,475,845), Lee (Adding External Input Sources to the X Toolkit Event Loop) and further in view of Connelly et al. (U.S. 5,706,515).

Art Unit: 2126

As to claim 5, APA does not teach the JAVA application comprises a plurality of application threads and a class defining a pair of JAVA methods having a first one of the JAVA methods that mirrors the first one of the functions that pauses execution of the X event loop for each application thread and a second one of the JAVA methods mirroring the second one of the functions that resumes execution of the X event loop for each application thread wherein the first and second JAVA methods prevent more than one application thread at a time from making a call to the visualization toolkit or widget. Connelly teaches using operation to allow only one thread can access the share resource in a multithread application (multiple threads of execution ... procedure 136; col. 2, line 53 – col. 3, line 64). It would have been obvious to apply the teaching of Connelly to the system of APA because it would provide a method to synchronize access of the share resource in a multithreading environment.

As to claim 6, APA does not teach each one of the application threads comprises a member variable of the class that is used in determining which one of the application threads can make a call to the visualization toolkit or widget. Connelly teaches a variable is used to determine which one of the application threads can make calls to the share resource (For each resource ... data structure; col. 3, line 31 – col. 4, line 14).

As to claim 7, APA does not teach each one of the application threads comprises a member variable of the class that is used to determine which one of the application threads executes thereby causing all of the remaining application threads to suspend execution. See rejection of claim 6 above. Connelly further teach while a thread is making call to the resource, the rest of the threads that want to make call to the resource are suspended.

Art Unit: 2126

As to claim 8, APA does not teach each one of the application threads comprises a member variable of the class that is used in restricting making a call to the visualization toolkit or widget to only one application thread at a time. See rejection of claim 6 above.

As to claim 9, APA does not teach the member variable can be set to (a) one of a pair of values with a first one of the values indicating that (i) one of the plurality of JAVA functions of one of the application threads has been called and (ii) the other one of the plurality of JAVA functions of the one of the application threads has not yet been called and (b) a second one of the values indicating that both of the JAVA functions of the one of the application threads have been called. APA teaches (state value; col. 3, lines 31- 64) the member value can be set to “signaled” or “unsigned”. It would have been obvious to apply the teaching of Connelly to the system of APA for the same reason as claim 6 above.

As to claim 10, APA does not teach the first one of the JAVA methods checks the member variable and issues a wait call suspending operation of the one of the application threads if the member variable has the first one of the values and permits execution of the one of the application threads to continue if the member variable has the second one of the value. Connelly teaches a Wait operation is used to implement an atomic wait (col. 2, line 60 – col. 3, line 19).

As to claim 11, APA does not teach the wait call comprises call to the standard JAVA Object wait() method. Connelly does not teach the Wait operation is the standard Java Object wait() method. It would have been obvious to override the Java object wait() method to provide more functionality of the method.

As to claim 12, APA does not teach the member variable is set to the second one of the values before completing execution of the one of the application threads. Connelly does not

Art Unit: 2126

explicitly teach the member variable is set to the second one of values before completing execution of the one of the application thread. Connelly teaches the member variable is set to either “signaled” or “unsignaled”. It would have been obvious to modify the teaching of Connelly and apply to the system of APA because it would help to provide an atomic wait function in a multithreading application.

As to claim 13, APA does not teach a call is made to the standard JAVA Object notifyAll upon completion of the one of the application threads. APA teaches a call is made to the standard JAVA Object notifyAll upon completion of the one of the application threads (Notify All operation; col. 3, lines 20-30 and col. 7, line 25 – col. 8, line 24).

As to claim 14, APA does not explicitly teach making a call to notifyAll upon completion of the one of the application threads invokes the first one of the JAVA methods to determine which one of the plurality of application threads should be executed. Connelly teaches making a call to notifyAll upon completion of the one of the application threads invokes the first one of the JAVA methods to determine which one of the plurality of application threads should be executed (Notify All operation; col. 3, lines 20-30 and col. 7, line 25 – col. 8, line 24).

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Diem K Cao whose telephone number is (703) 305-5220. The examiner can normally be reached on Monday - Friday, 9:00AM - 5:00PM.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-3900.

Any response to this action should be mailed to:

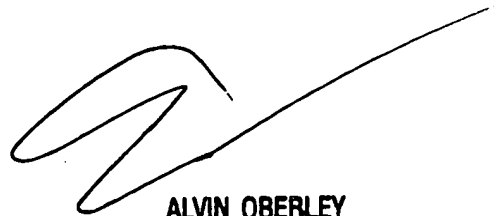
Art Unit: 2126

Commissioner of Patents and Trademarks
Washington, DC 20231

Or fax to:

- AFTER-FINAL faxes must be signed and sent to (703) 746-7238.
- OFFICIAL faxes must be signed and sent to (703) 746-7239.
- NON-OFFICIAL/DRAFT faxes should not be signed, please send to (703) 746-7140.

Diem Cao
March 24, 2003

A handwritten signature in black ink, consisting of a large, stylized 'A' followed by a long, sweeping horizontal line that extends to the right.

**ALVIN OBERLEY
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100**